ERDC MSRC/PET TR/00-22

Parallel Software Tools and the Parallel Performance of the CE-QUAL-ICM Water Quality Simulator

by

Mary F. Wheeler
Victor J. Parr

16 June 2000

07h0032000

Parallel Software Tools and the Parallel Performance of the CE-QUAL-ICM Water Quality Simulator

Mary F. Wheeler
and
Victor J. Parr
Center for Subsurface Modeling
University of Texas, Austin

June 16, 2000

1 Introduction

This report describes the work done in the focused effort to maintain and improve the performance of the parallel EQM programs developed by the PET EQM team at the University of Texas in Austin.

CE-QUAL-ICM was used to run 130 EPA senarios of Chesapkeake Bay last year at ERDC. These runs averaged about 12 wall clock hours each, and were run on 32 Cray T3E processors. Much larger model runs are planned in the near future. Therefore, the EQM team has profiled the parallel performance of CE-QUAL-ICM using both hardware vendor supplied profiling tools and those supplied by the PET Parallel Tools team.

2 Profiling with VAMPIR

CE-QUAL-ICM has been parallelized using MPI. There are two important aspects of the program, which directly affect scalability: the efficiency of the message-passing interface, and the efficiency of the file I/O.

The parallel tool VAMPIR has been used to profile both aspects. With the encouragment of Shirley Browne from University of Tennessee and with the technical assistance of Clay Breshears from Rice University, CE-QUAL-ICM has been profiled for a 30-day Chesapeake Bay scenario run.

3 Profiling the MPI Message-Passing Interface

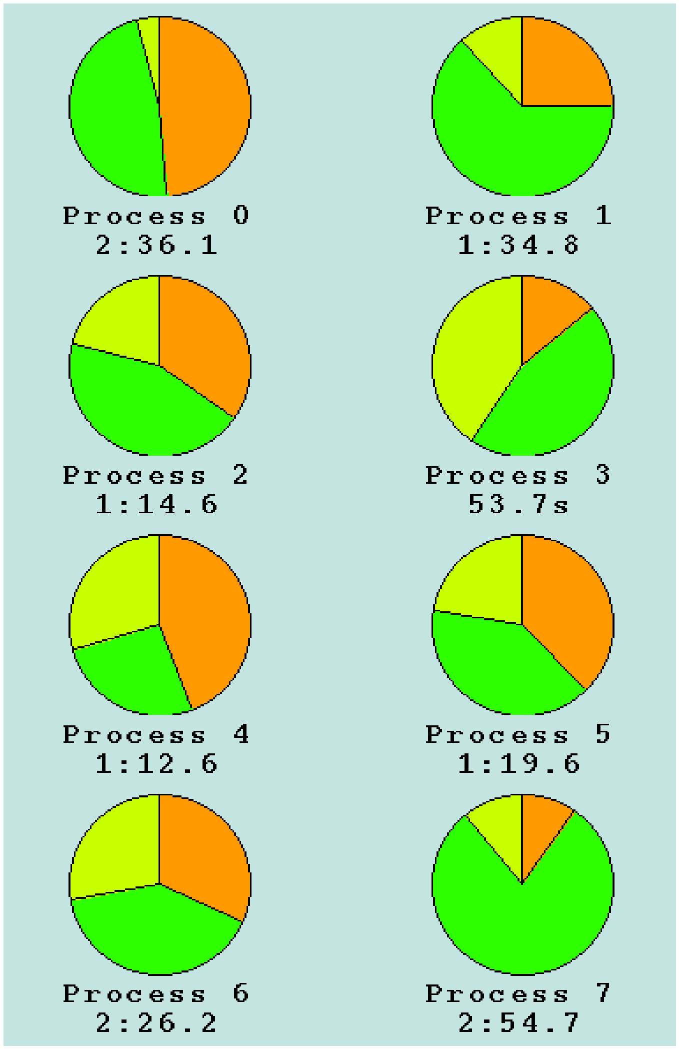
Figure 1 shows eight pie charts of the cpu time on each of eight processors spent in communicating the cell concentrations (light green wedges), cell volumes (brown wedges), and a collective all-reduce operation (dark green wedges) to determine the length of the next time-step to preserve numerical stability globally over the entire computational domain.

4 Asynchronous Communication of Concentration Data

The communication of the concentrations of the active constituents belonging to ghost cells of neighboring subdomains involves the largest volume of data to be communicated. The messages passed are on the order of 256,000 bytes per message. These messages are passed every timestep.

Figure 1 shows, however, that the cpu time spent in communicating this data (light green wedges) is rather minimal. This is due to the fact that asynchronous sends and receives are used with the fortunate fact that the data is not needed for computation immediately. This situation allowed the asynchronous sends and receives to be initiated at the beginning of the timestep loop, and the wait for completion to be delayed until all significant work in the timestep loop is complete, including writing out to all files.

Figure 1 CPU times for Message Passing on 8 Processors



5 Rethinking the Volume Communication

The communication of the cell volumes, since cell volumes change as the water levels go up and down, is not such a fortunate situation, because the updated cell volumes are needed immediately.

After much discussion among the team, and careful analysis of the data dependencies involving the cell volumes, it was decided to add some logic to the localization of the hydrodynamic input file, to write not only volumes to cells which are resident to a subdomain, but to write volumes for ghost cells as well, so that each subdomain contains all the volume information that it needs, and therefore, no message-passing is required. This technique did not occur to us when the original pre-processor to the parallel version of CE-QUAL-ICM was developed. However, the VAMPIR profiling provided the insight that parallel performance, and therefore, scalability is hindered by the way the cell volume updates were treated.

6 Timestep calculation

As the VAMPIR pie charts (Figure 1) show clearly, the remaining problem is the timestep calculation (the dark green wedges), which was coded as the collective operation MPI_ALLREDUCE.

This collective operation is implemented efficiently by the MPI library using a "log n" approach to the message passing involved in gathering an estimate of the next timestep from each processor by the root processor, followed by a broadcast of the minimum of these numbers (which is a stable timestep for all subdomains).

The problem is that the processors get out of phase with each other, because of waiting on I/O at the bottom of the timestep loop, and arrive at the collective operation at different times, and the operation cannot begin until every processor has arrived at the synchronization point.

The solution to this problem may be found through rewriting the collective operation as a sequence of asynchronous operations, which will remove the synchronization point, and allow hiding the message passing within required computation.

The EQM team analyzed this approach and found it very efficient for CE-QUAL-ICM, but a little tricky to code.

7 Persistent Messasge Passing

All of the parallel EQM codes developed for PET thus far have the property that each processor passes the same information to its neighbors each timestep. But the MPI library using the standard MPI_ISEND and MPI_IRECV routines do a lot of setup each time called.

A chance conversation with Jeff Squyres at Notre Dame, led to the use of the *persistent* message-passing routines, where the message passing protocol and data structures are established at the beginning of the run, and a simple *STARTALL* initiates the communication.

The EQM teacm first implemented this method in the f90 version of ADCIRC, with a large improvement in ADCIRC's scalability (120 speedup on 128 processors). The EQM team plans to include this in the rewrite of the CE-QUAL-ICM message-passing interface.

8 Improving the Scalability of File I/O

The VAMPIR profile of CE-QUAL-ICM did not show I/O to be a problem for several reasons. Firstly, the output files are not written very often, and the benchmark was cut off after 30 days of simulation, because a longer run crashed after VAMPIR had collected so much data in core that the heap space overflowed the job space. The second reason is that the benchmark only used eight preocessors.

However, for CE-QUAL-ICM to scale efficiently for larger proposed EPA work, the *simultaneously open file* problem inherent in our approach to parallelizing EQM programs must be addressed. The problem is that the number of files in the parallel program is equal to the product of the number of files in the original program with the number of processors used in the run. For CE-QUAL-ICM, the Chesapeake Bay EPA scenario runs have 13 files open simultaneously per processor. For a 100 processor run the total number of files open in the job would be 1300 files. The large number of open files inherent in our parallelization scheme presents a challenge to scalability and to I/O performance, and is a problem because of limitation imposed by the installed operating systems.

Feedback from the users of the parallel version of CE-QUAL-ICM indicates that for long 10 year simulations it takes a few hours for the post-processor to globalize the local output files created by each processor, which are local to a subdomain of the whole computational domain.

Shirley Browne at the University of Tennessee has suggested a solution to this problem, which has two good side effects. She has suggested using MPI-2 (which has been installed on the Cray T3E at ERDC MSRC), and its asynchronous shared file routines. This technique will greatly reduce the number of open files, since all of the local output files which correspond to a single file in the original serial program will again be fused into a single global output file. We estimate this will reduce the number of simultaneously open file for a Chesapeake Bay scenario run by about 500 files. In addition, the asynchronous file I/O will reduce I/O wait time.

The EQM team has decided to explore this avenue by using MPI-2 shared files for output, so that global files are written during simulation. This will make the output data available at runtime. Conversations with Carl Cerco and Barry Bunch revealed that in the case of CE-QUAL-ICM the raw data is not immediately usable, but must be run through post-processing software for interpretation. However, in the hydrodynamic case, with codes such as CH3D, RMA-10, and ADCIRC, this would be useful for real-time applications such as flood prediction.

9 Software Development Plan

Using the above analysis the EQM team has decided to implement a rewrite of the message-passing for CE-QUAL-ICM, which eliminates message-passing altogether for the volume updates, uses an asynchronous coding of the timestep update, and uses persistent message-passing routines for all of the message-passing. Moreover, the asynchronous shared file routines in MPI-2 will be used to write all the output files as global files and phase out the post-processor for CE-QUAL-ICM, which will have the two side effects of (1) increased parallel scalability and (2) reduction in software maintenance for CE-QUAL-ICM, which changes from year to year as new projects arise.

10 Conclusions

Both VAMPIR and MPI-2 can greatly benefit the scalability of EQM codes, which are used for production environmental work. We believe that the techniques discussed here are valuable for all of the parallel EQM codes developed thus far. Finally, we thank the PET Parallel Tools people for training our team on the usage and utility of these tools.